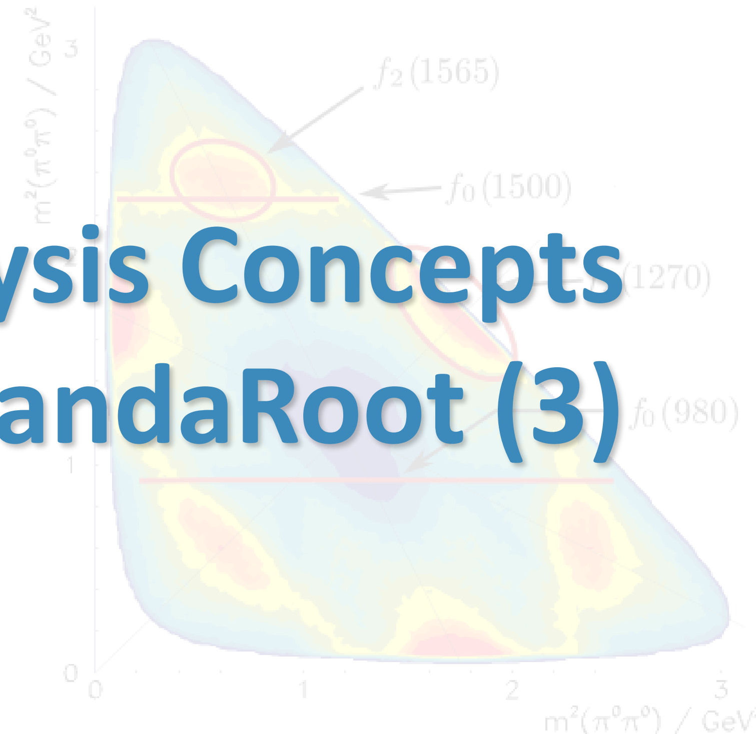
Physics Analysis Concepts with PandaRoot (3)



PANDA Lecture Week 2017

GSI, Dec 11 - 15, 2017

Klaus Götzen
GSI Darmstadt

Topics

- Effective Analysis - Working with ROOT TTrees

- Event Generators:

  EvtGen, DPM, FTF, Box Generator

- Generation and Simulation
  - Stand-alone generation

    (*simpleEvtGen, DPMGen, FTFGen*)
  - Fast Simulation (*PndFastSim, PndFsm*...)

EFFECTIVE ANALYSIS ROOT TREES

Analysis with ROOT TTree

- User analyses commonly use histograms to store results like invariant mass distributions/angles

- This has several disadvantages
  - Can't change to finer binning (detect small scale features)
  - Can't change scope (extend range to larger ROI)
  - Can't check impact of related cuts
  - Can't study correlations between observables
  - Can't study additional variables not put to histograms

- My personal recommendation

  - Store **TTree** instead of (or in addition to) histograms
  - This overcomes all issues above

Analysis with ROOT TTree

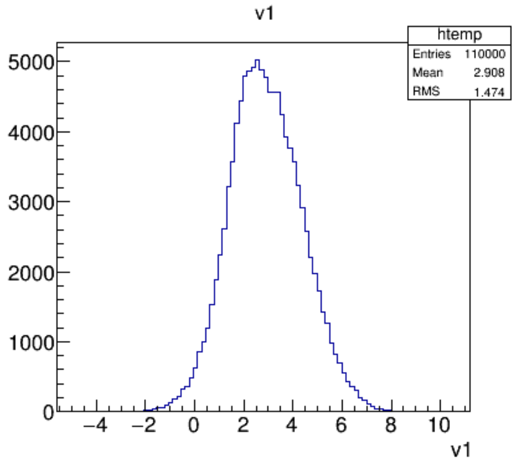
- ROOT's TTree
  - Imagine as a large table with a row per event/combination
  - Columns are so-called branches containing the values
  - Offers simple interactive analysis
  - Most powerful command-line feature **TTree::Draw**

```
> root ntp0_dd.root
root [0]
Attaching file ntp0_dd.root as _file0...
root [1] .ls
TFile**         ntp0_dd.root    chain files
 TFile*         ntp0_dd.root    chain files
  KEY: TTree    ntp0;1  D0->K- pi+
root [2] ntp0->GetEntries()  // get number of events/entries in TTree
(Long64_t) 1710036
root [3] ntp0->Print()       // get information about branches
******************************************************************************
*Tree    :ntp0      : D0->K- pi+                                             *
*Entries :  1710036 : Total =      1006429833 bytes  File  Size =  573205551 *
*        :          : Tree compression factor =   1.76                      *
******************************************************************************
*Br    0 :ev        : ev/I                                                   *
*Entries :  1710036 : Total  Size=    6846211 bytes  File Size  =    1994787 *
*Baskets :       65 : Basket Size=     391168 bytes  Compression=   3.43     *
*............................................................................*
*Br    1 :cand      : cand/I                                                 *
...
```

TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

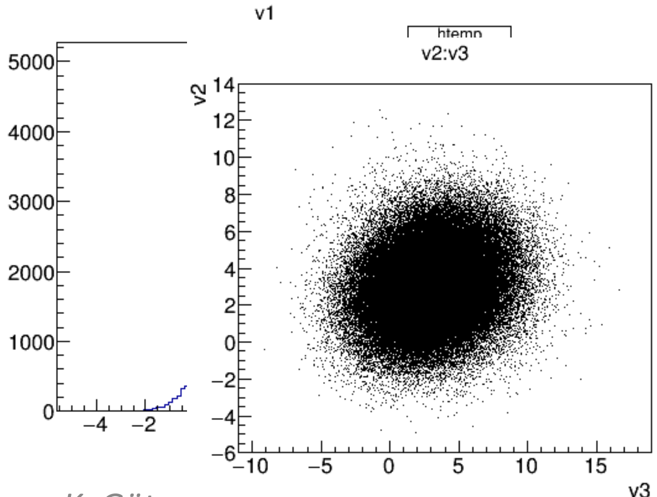
```
root [0] tree->Draw("v1")              // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")           // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")       // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")   // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")   // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")  // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)    // first 100 entries of v3
```



TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

```
root [0] tree->Draw("v1")                 // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")              // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")          // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")      // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")      // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")     // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)       // first 100 entries of v3
```

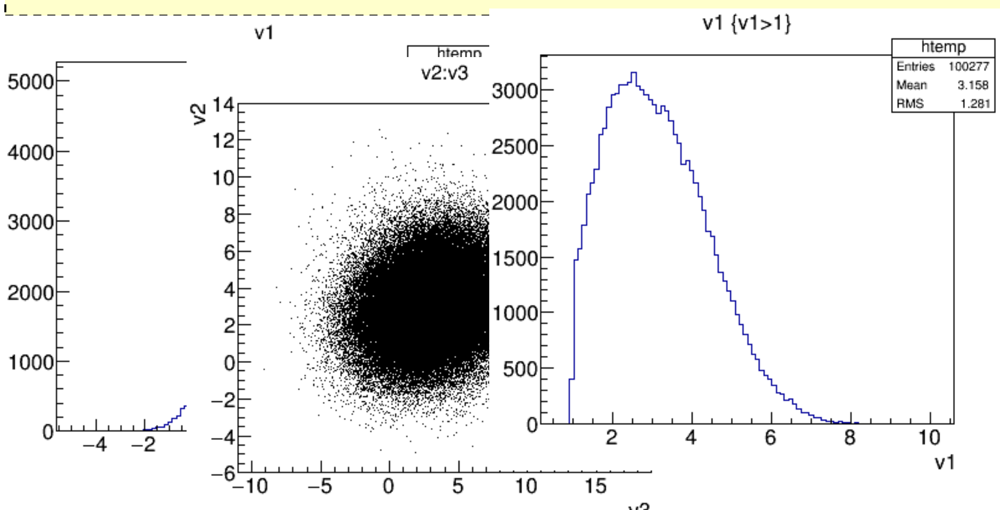


TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`

- Some examples:

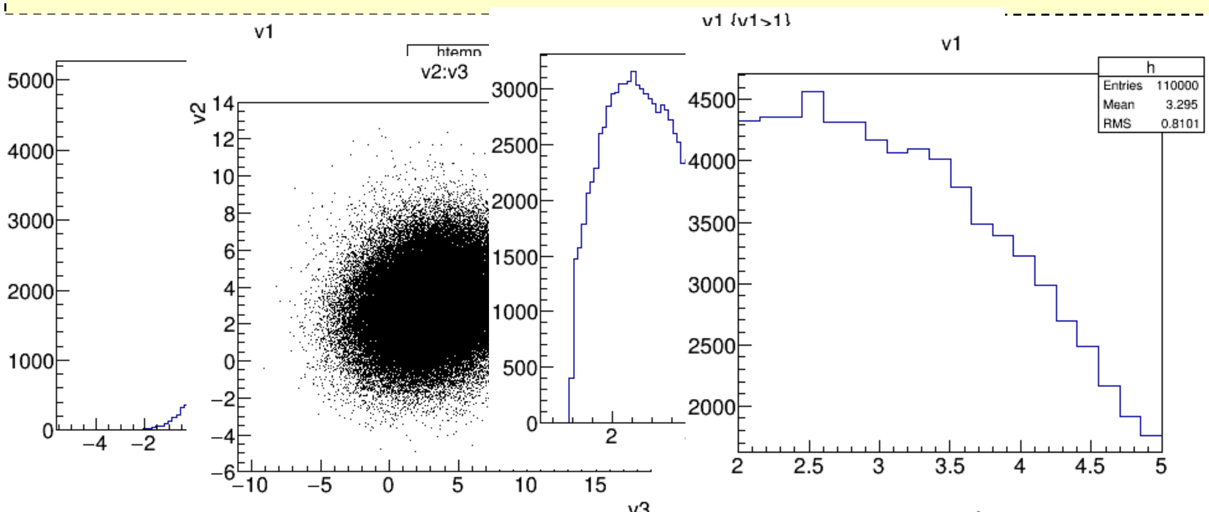
```
root [0] tree->Draw("v1")              // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")           // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")       // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")   // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")   // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")  // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)    // first 100 entries of v3
```



TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

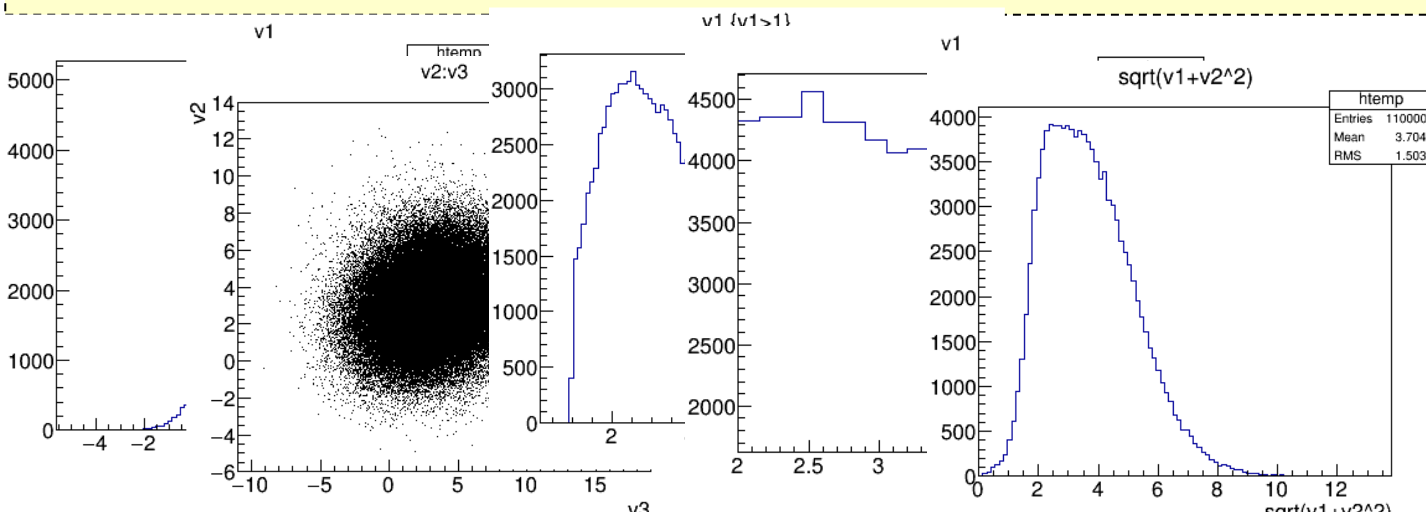
```
root [0] tree->Draw("v1")                // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")             // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")         // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")     // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")     // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")    // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)      // first 100 entries of v3
```



TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

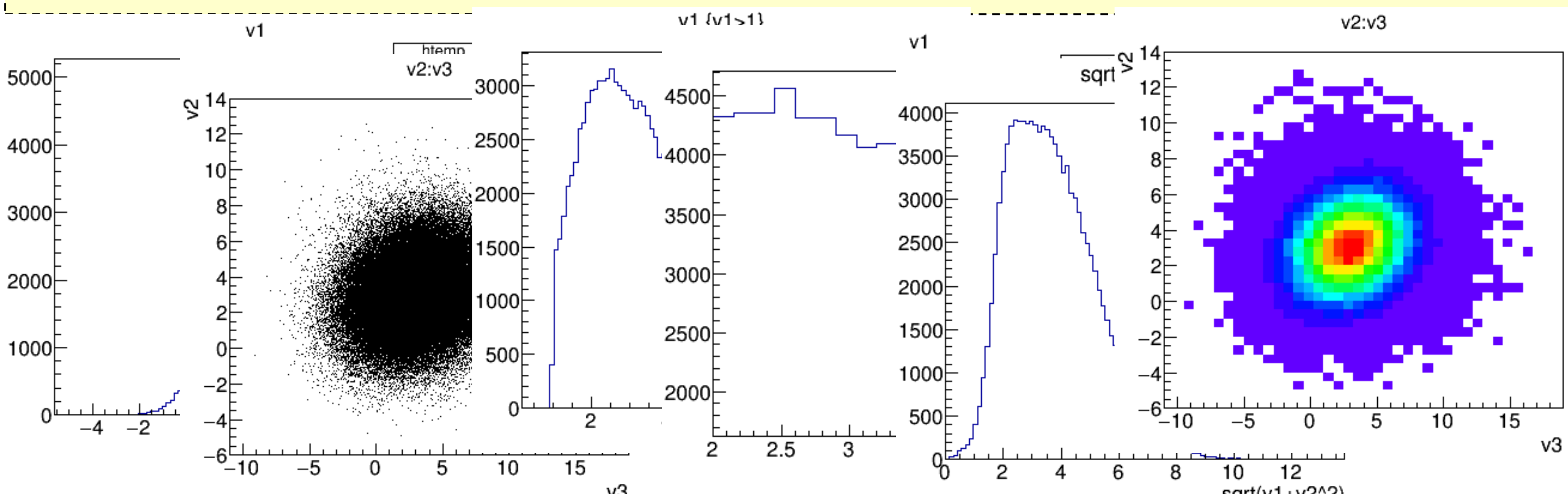
```
root [0] tree->Draw("v1")                 // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")              // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")          // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")      // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")      // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")     // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)       // first 100 entries of v3
```



TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

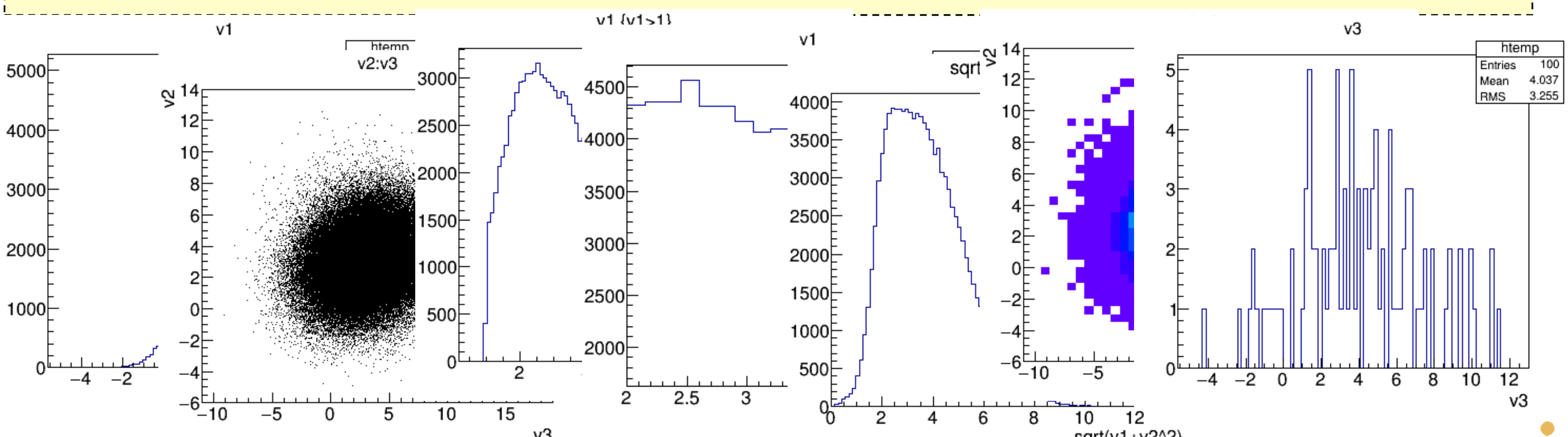
```
root [0] tree->Draw("v1")                 // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")              // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")          // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")      // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")      // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")     // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)       // first 100 entries of v3
```



TTree::Draw Basics

- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

```
root [0] tree->Draw("v1")               // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")            // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")        // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")    // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")    // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")   // 2D histogram with color map instead scatter plot
root [6] tree->Draw("v3","","",100)     // first 100 entries of v3
```



TTree::Draw Basics

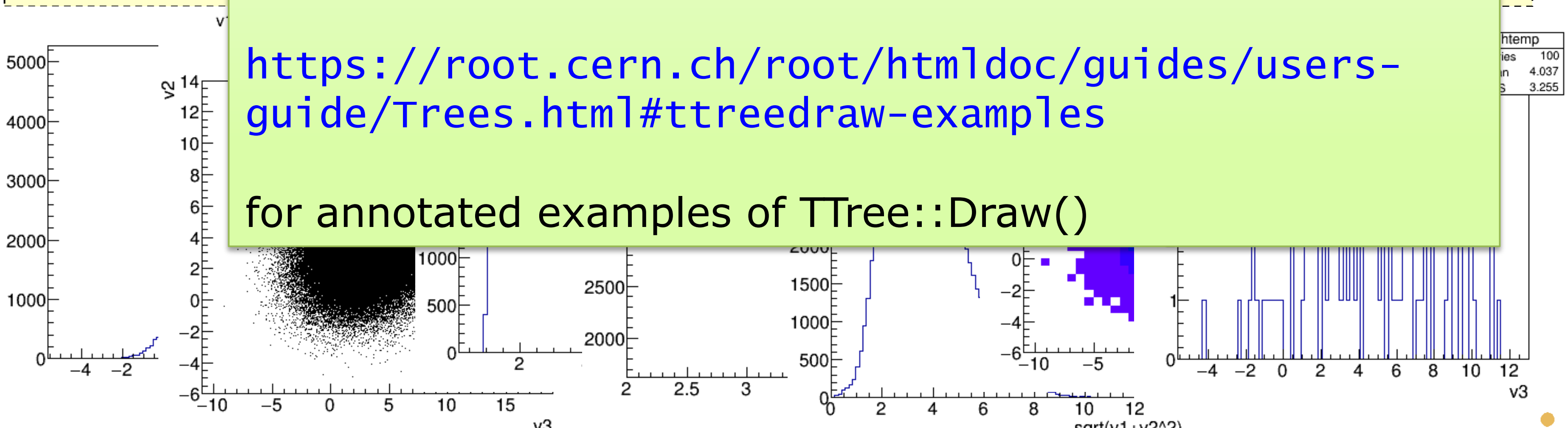
- TTree::Draw syntax
  `t->Draw(expression(s) [,cut,option,nentries,firstentry])`
- Some examples:

```
root [0] tree->Draw("v1")                 // distribution of variable 'v1' (auto-range histo)
root [1] tree->Draw("v2:v3")              // 'v1' vs. 'v2' -> correlation
root [2] tree->Draw("v1","v1>1")          // 'v1' only for cases when 'v1>1'
root [3] tree->Draw("v1>>h(20,2,5)")      // 'v1' in histogram with 20 bins, range [2..5]
root [4] tree->Draw("sqrt(v1+v2^2)")      // result of formula for each event (auto-range)
root [5] tree->Draw("v2:v3","","col")     // 2D histogram with color map instead scatter plot
root [6] tree
```

Take a look to

https://root.cern.ch/root/htmldoc/guides/users-guide/Trees.html#ttreedraw-examples

for annotated examples of TTree::Draw()



EVENT/PARTICLE GENERATORS

Important Particle Generators in PANDA

- EvtGen (PndEvtGenDirect / simpleEvtGen)
  - Generate signal reactions or specific backgrounds
- DPM - Dual Parton Model (PndDpmDirect / DPMGen)
  - Study $\bar{p}p$ background reactions
- FTF (PndFtfDirect / FTFGen)
  - Study $\bar{p}p$ and $\bar{p}A$ background reactions
- Particle Gun (PndBoxGenerator)
  - Single tracks for acceptance, efficiency, resolution studies
- Many others not discussed here
  - GiBuu, UrQmd, Fluka, Pythia,...

EvtGen - Properties

- **EvtGen** - Generating specific signal/background reactions
  - Decayer rather than generator
  - Knows many actual particle properties (`evt.pdl`) and decays (`DECAY.DEC`)
  - Extendable with new decay models
  - Configuration of a well defined decay chain

```
noPhotos

Decay pbarpSystem
  1.0     J/psi pi+ pi-    PHSP;
Enddecay

Decay J/psi
  0.0597  e+  e-           VLL;
  0.0596  mu+ mu-          VLL;
Enddecay

End
```

EvtGen - Properties

- **EvtGen** - Generating specific signal/background reactions
  - Decayer rather than generator
  - Knows many actual particle properties (`evt.pdl`) and decays (`DECAY.DEC`)
  - Extendable with new decay models
  - Configuration of a well defined decay chain

```
noPhotos

Decay pbarpSystem
  1.0      J/psi pi+ pi-     PHSP;
Enddecay

Decay J/psi
  0.0597  e+  e-            VLL;
  0.0596  mu+ mu-           VLL;
Enddecay

End
```

EvtGen switches; Alias definitions

Decay definition

Decay Models

Decay fractions (autom. normalized)

End of decay file with <CR>

EvtGen - Particles and Decays

- Knowledge base: particle list, decay table, decay models

evt.pdl

```
* 5/10/2013 Updated by R. Godang. The format and convention are based on the current evt.pdl and PDG 2012
*------------------------------------------------------------------------------------------------------------
*                 name                 id       mass/GeV    width/GeV     max_Dm/GeV   3*charge  2*spin   lifetime*c/mm    PythiaId
add  p Particle  K_4*+                         329  2.0450000e+00  1.9800000e-01  2.0000000e-01    3     8  0.0000000e+00        0
add  p Particle  h_b(2P)                    110553  1.0255000e+01  0.0000000e+00  0.0000000e+00    0     2  0.0000000e+00   110553
add  p Particle  b                               5  5.0000000e+00  0.0000000e+00  0.0000000e+00   -1     1  0.0000000e+00        5
add  p Particle  anti-nu_e                     -12  0.0000000e+00  0.0000000e+00  0.0000000e+00    0     1  0.0000000e+00      -12
add  p Particle  D_2*0                         425  2.4611000e+00  4.3000001e-02  3.0900000e-01    0     4  0.0000000e+00      425
add  p Particle  Upsilon                       553  9.4603000e+00  5.4000022e-05  5.0000000e-04    0     2  0.0000000e+00      553
add  p Particle  anti-B'_10                 -20513  5.7570000e+00  2.5027080e-01  2.0000000e-01    0     2  0.0000000e+00   -20513
add  p Particle  anti-K0                      -311  4.9761400e-01  0.0000000e+00  0.0000000e+00    0     0  0.0000000e+00     -311
add  p Particle  Lambda(1405)0               13122  1.4060000e+00  5.0000015e-02  7.0000000e-02    0     1  0.0000000e+00        0
add  p Particle  K*+                           323  8.9166000e-01  5.0800012e-02  2.3000000e-01    3     2  0.0000000e+00      323
add  p Particle  e+                            -11  5.1099891e-04  0.0000000e+00  0.0000000e+00    3     1  0.0000000e+00      -11
```

DECAY.DEC

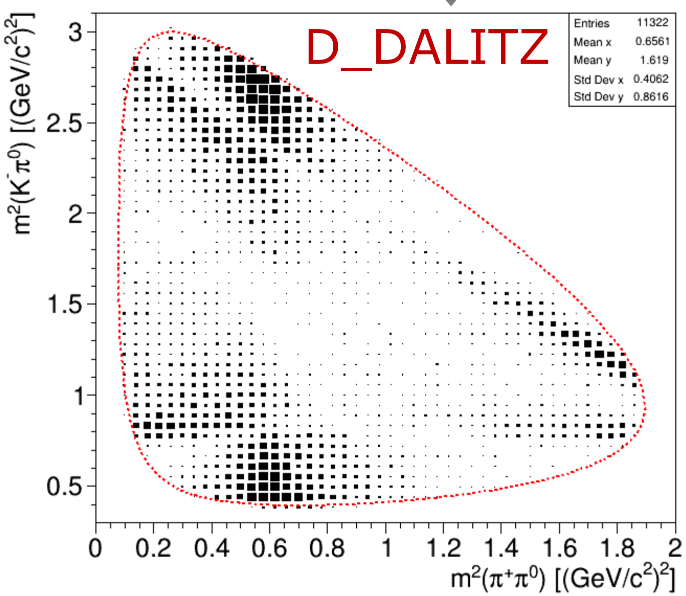
```
#
Decay anti-K_0*0
0.6667      K-   pi+                     PHSP;
0.3333      anti-K0    pi0               PHSP;
Enddecay
#
Decay anti-K_0*0N
1.0000      anti-K0    pi0               PHSP;
Enddecay
#
Decay K_10
0.2800    rho-   K+                   VVS_PWAVE 1.0 0.0 0.0 0.0 0.0 0.0;
0.1400    rho0   K0                   VVS_PWAVE 1.0 0.0 0.0 0.0 0.0 0.0;
0.1067    K*+    pi-                  VVS_PWAVE 1.0 0.0 0.0 0.0 0.0 0.0;
0.0533    K*0    pi0                  VVS_PWAVE 1.0 0.0 0.0 0.0 0.0 0.0;
```

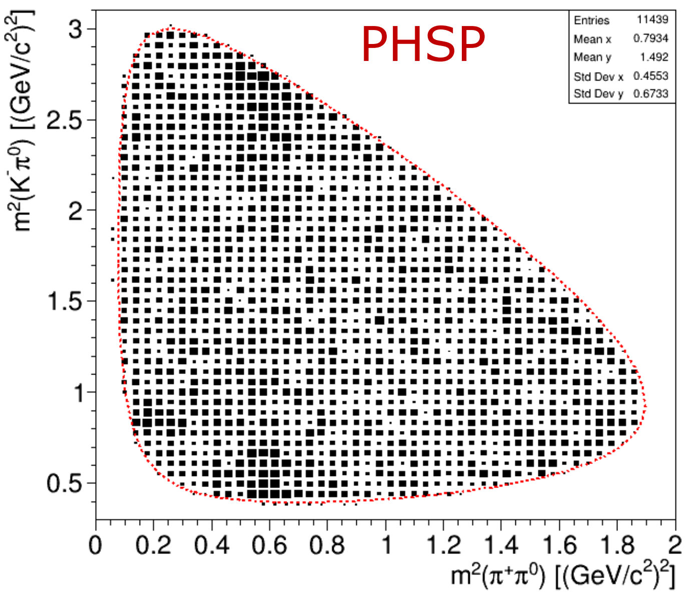
EvtGen - Aliases and Decay-Models

```
Alias    MyD0    D0
Alias    MyD0b   anti-D0

Decay pbarpSystem
   0.5    MyD0  anti-D0    PHSP;
   0.5    MyD0b  D0        PHSP;
Enddecay

Decay MyD0
   1.0    K- pi+ pi0       PHSP;
Enddecay

Decay MyD0b
   1.0    K+ pi- pi0       D_DALITZ;
Enddecay

End
```

Alias is copy of known particle

Decay not specified
→ decay according to DECAY.DEC!

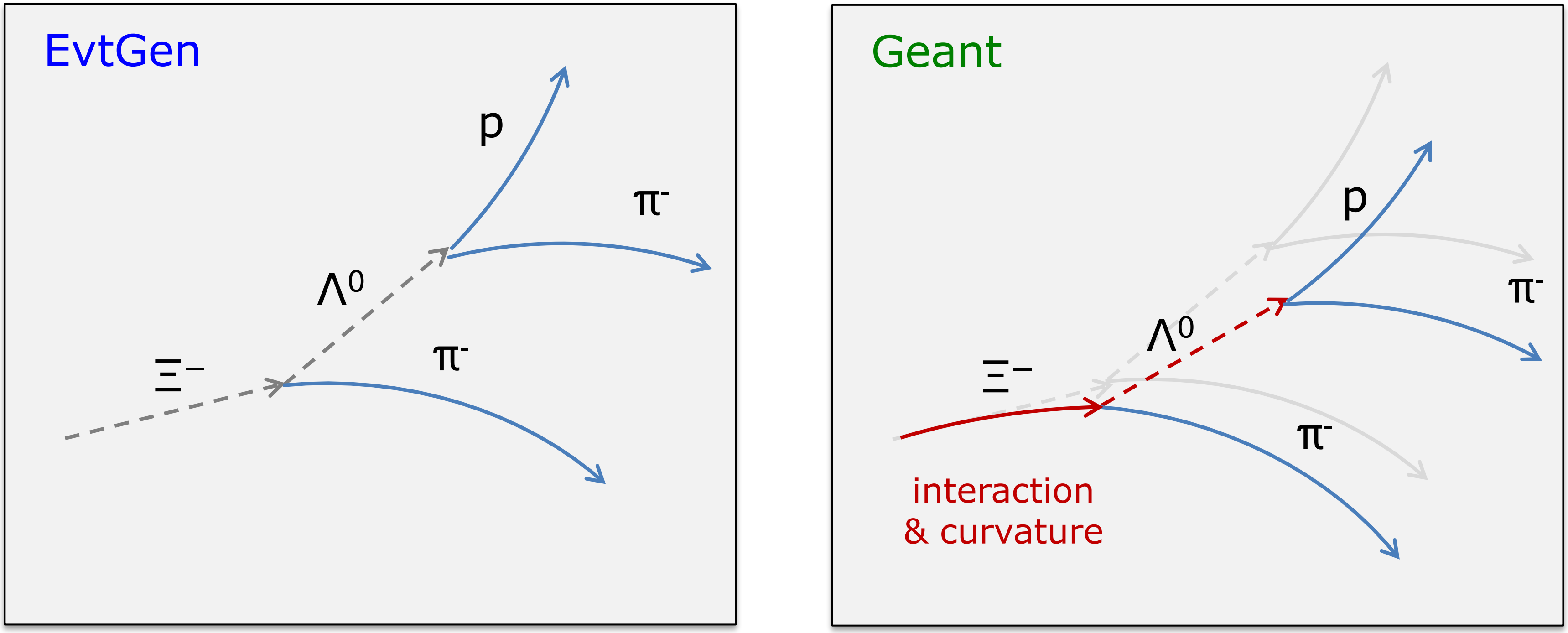




EvtGen - CAVEAT: Long Living Particles

A word of warning ...

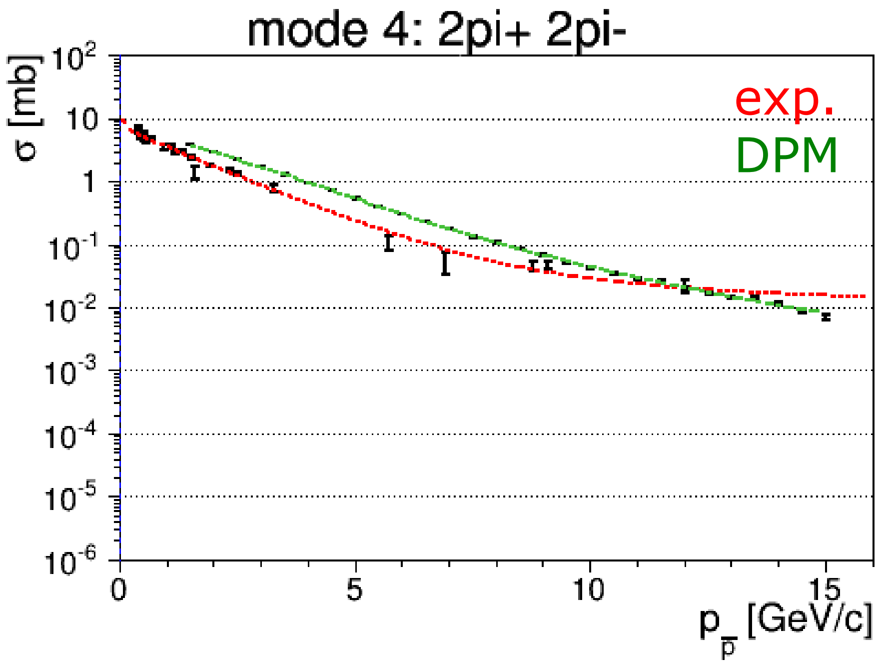
- Definition of long-living (charged) particle decays of e.g. Ξ^+ or Σ^+ prevents proper simulation of interaction and curvature

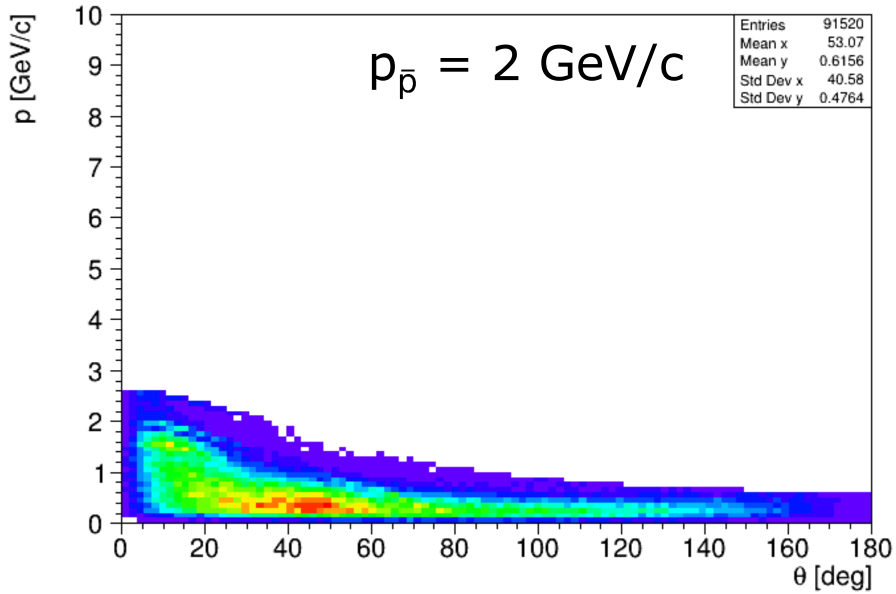


- Partial workaround:
https://panda-wiki.gsi.de/foswiki/pub/Computing/Minutes02May2017/2.5.2017_teammeeting.pdf

DPM (Dual Parton Model) - Studying Background

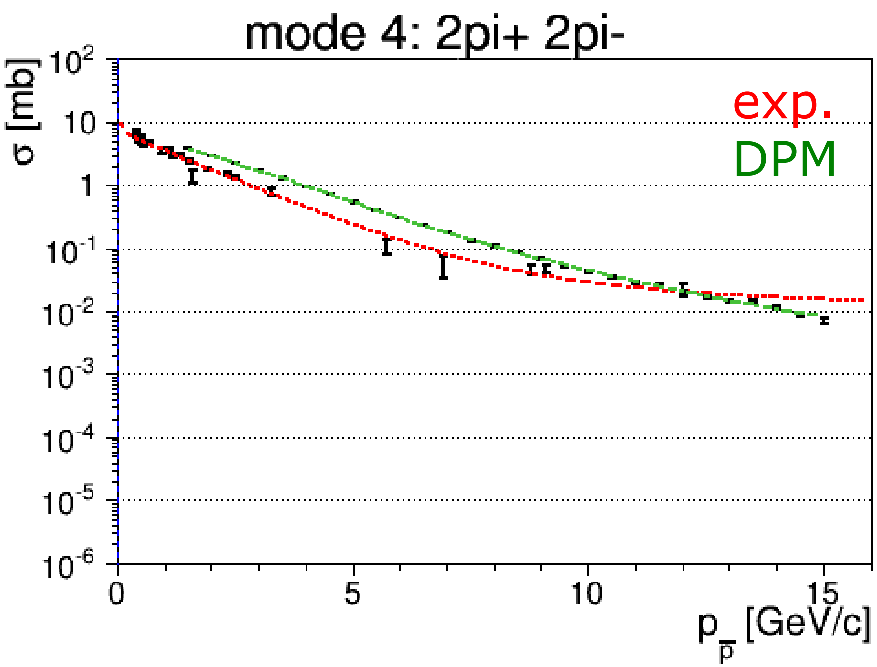
- **DPM** - Generating generic background
  - Produces generic $\bar{p}p$ reactions (simulates hadronization)
  - Study multiplicities, phsp coverage, background level
  - Roughly reproduces generic cross sections
  - Intermediate resonances (decay switchable)
    - No displacement for decaying long living resonances!

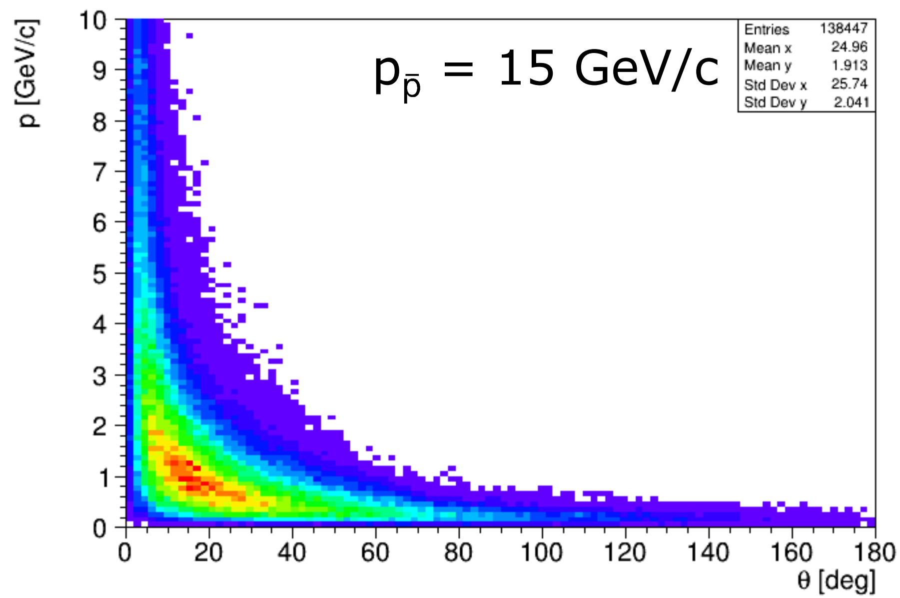




DPM (Dual Parton Model) - Studying Background

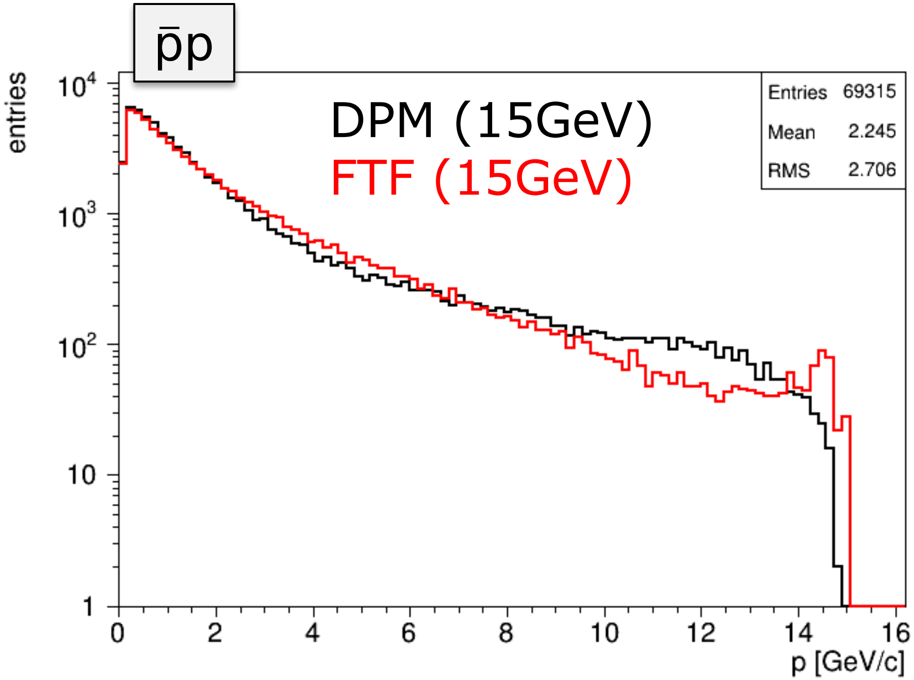
- **DPM** - Generating generic background
  - Produces generic $\bar{p}p$ reactions (simulates hadronization)
  - Study multiplicities, phsp coverage, background level
  - Roughly reproduces generic cross sections
  - Intermediate resonances (decay switchable)
    - No displacement for decaying long living resonances!

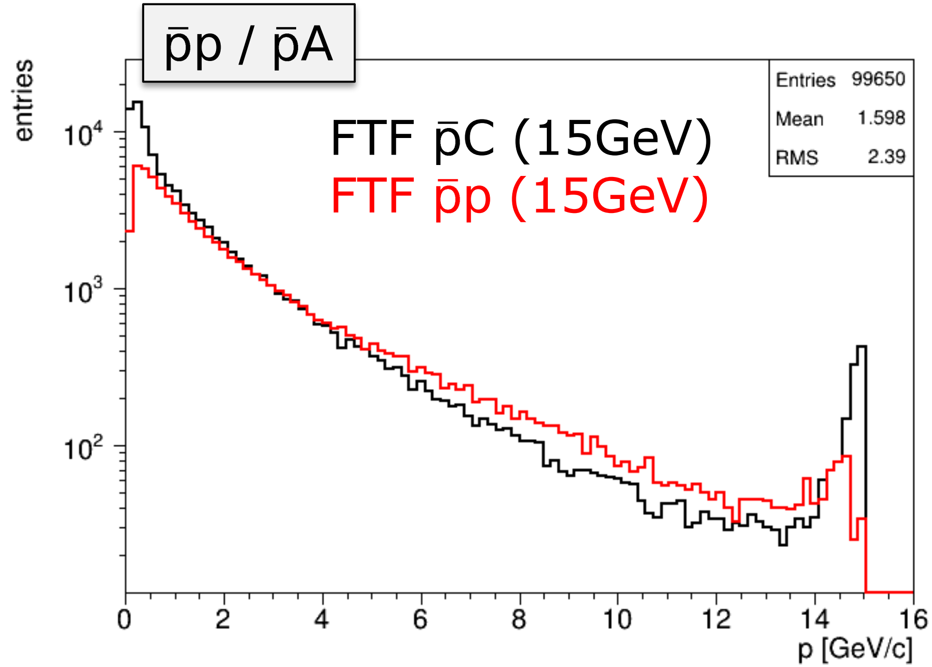




FTF - Background with proton and nuclear targets

- **FTF** - Generating generic background with p **and A** targets
  - Produces generic p̄p/p̄A reactions
  - As DPM: multiplicities, phsp coverage, background level
  - Intermediate resonances
  - Compares reasonably to DPM





Insertion: Online $\bar{p}p$ Cross Section DB

- Estimating cross sections for particular final states
  → https://panda.gsi.de/pbarx/

$\bar{p}p$ cross section estimator (data version)

Determines main possible background channels being similar to signal final state for certain E_{cm}, sorted by cross section, based on experimental data.

Fill in final state and the E_{cm} (negative values = $-p_{pbar}$) you are interested in and hit 'search'. Results are sorted according to their cross section values taken from the
sometimes fits are not perfect; check plots on bottom of the page).

Direct links: [Total] [Elastic] [Inelastic vs. sum of channels]

- **Final State**: Final state of interest. Empty field lists all modes.
  Particles: e+, e-, mu+, mu-, pi+, pi-, K+, K-, p, pb, pi0, eta, gamma, KS, KL, n0, n0b (prefix numbers allowed)
- **Exact**: Only lists modes comprising all final states with correct type.
- **E_{cm}**: Center-of-mass energy. Negativ values = $-p_{pbar}$.
- **+chrg / +neut**: Maximum number of additionally allow charged / neutrals in a background reaction to be listed.
- **Cut**: Required x-section ratio of weakest to stronges mode to be listed.

Channel

Final State [pi+ pi- pi0] ☐ Exact E_{cm} [2.5] +chrg [2] +neut [2] Cut [0.01] [search]

Switch to DPM version

signal: pi+ pi- pi0 (2 chrg, 2 neut, p_{thresh} = 0.000 GeV/c) @ E_{cm} = 2.500 GeV / p_{pbar} = 2.201 GeV/c

additional charged allowed : 2
additional neutrals allowed : 2

Possible background channels ($sigma_{tot}$ = 23203.036 mub) are:

n	idx	mode	xs [mub]	p_{thres} [GeV/c]
1	6	2pi+ 2pi- 2pi0	7393.145	0.000
2	5	2pi+ 2pi- pi0	7040.281	0.000
3	30	pi+ pi- 2pi0	3663.379	0.000
4	17	pb p pi0	2434.973	0.777
5	14	om pi+ pi-	1091.377	0.000

Insertion: Online p̄p Cross Section DB

- Estimating cross sections for particular final states
  → https://panda.gsi.de/pbarx/

p̄p cross section estimator (data version)

Determines main possible background channels being similar to signal final state for certain E_{cm}, sorted by cross section, based on experimental data.

Fill in final state and the E_{cm} (negative values = $-p_{pbar}$) you are interested in and hit 'search'. Results are sorted according to their cross section values taken from the
sometimes fits are not perfect; check plots on bottom of the page).

Direct links: [Total] [Elastic] [Inelastic vs. sum of channels]

- **Final State**: Final state of interest. Empty field lists all modes.
  Particles: e+, e-, mu+, mu-, pi+, pi-, K+, K-, p, pb, pi0, eta, gamma, KS, KL, n0, n0b (prefix numbers allowed)
- **Exact**: Only lists modes comprising all final states with correct type.
- **E_{cm}**: Center-of-mass energy. Negativ values = $-p_{pbar}$.
- **+chrg / +neut**: Maximum number of additionally allow charged / neutrals in a background reaction to be listed.
- **Cut**: Required x-section ratio of weakest to stronges mode to be listed.

final state

Channel

Final State: pi+ pi- pi0 ☐ Exact E_{cm}: 2.5 +chrg 2 +neut 2 Cut 0.01 search

E_{cm}

additional particles allowed

Switch to DPM version

```
signal: pi+ pi- pi0 (2 chrg, 2 neut, p_thresh =  0.000 GeV/c) @ E_cm = 2.500 GeV / p_pbar =  2.201 GeV/c

additional charged allowed  :  2
additional neutrals allowed :  2
```

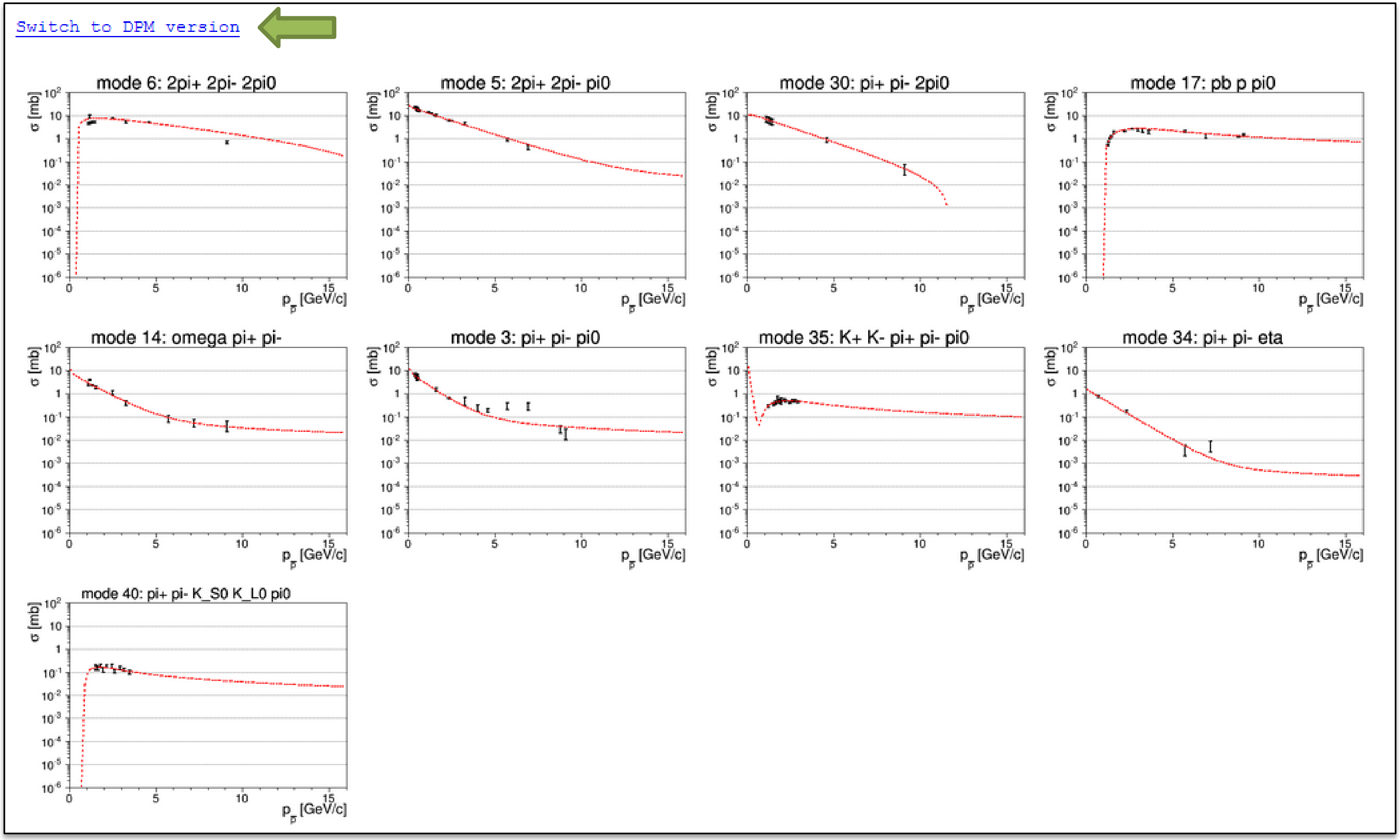
Possible background channels ($sigma_{tot}$ = 23203.036 mub) are:

n	idx	mode	xs [mub]	p_{thres} [GeV/c]
1	6	2pi+ 2pi- 2pi0	7393.145	0.000
2	5	2pi+ 2pi- pi0	7040.281	0.000
3	30	pi+ pi- 2pi0	3663.379	0.000
4	17	pb p pi0	2434.973	0.777
5	14	om pi+ pi-	1091.377	0.000

x-sections at that cm energy

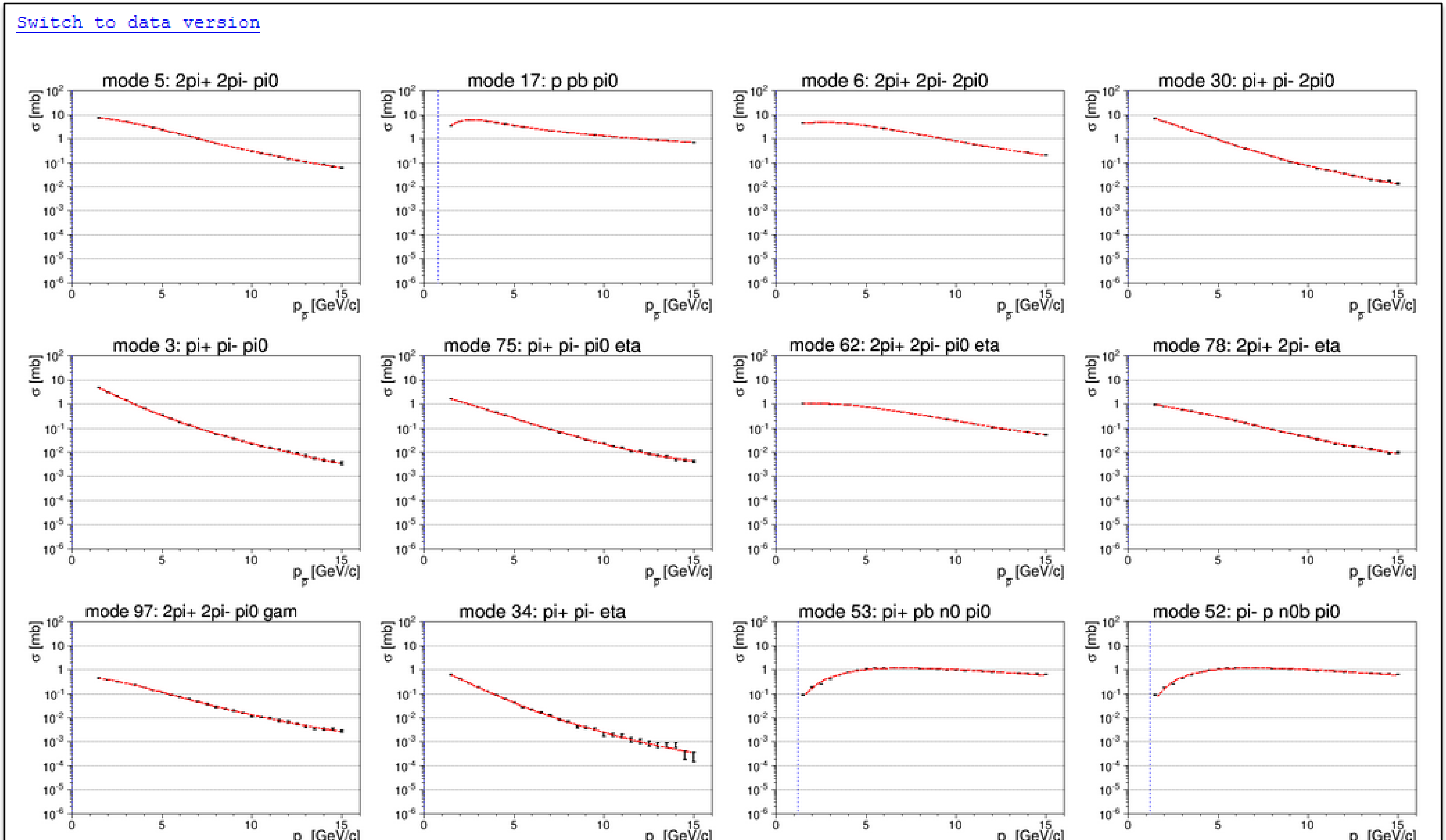
Insertion: Online p̄p Cross Section DB

- Estimating cross sections for particular final states
  → https://panda.gsi.de/pbarx/



Insertion: Online p̄p Cross Section DB

- Estimating cross sections for particular final states
  → https://panda.gsi.de/pbarx/



Insertion: Online $\bar{p}p$ Cross Section DB

- Example: Search for glue ball $G \rightarrow K^+ K^- \gamma$ @ 3.1 GeV (J/ψ energy)
- Possible backgrounds: $\pi^+\pi^-\pi^0$, $\pi^+\pi^-2\pi^0$, $2\pi^+2\pi^-\pi^0$, $2\pi^+2\pi^-2\pi^0$,...

Channel

Final State pi+ pi- pi0 ☑ Exact E_{cm} 3.1 +chrg 2 +neut 2 Cut 0.01 search

Switch to DPM version

```
signal: pi+ pi- pi0 (2 chrg, 2 neut, p_thresh =  0.000 GeV/c) @ E_cm = 3.100 GeV / p_pbar =  4.076 GeV/c

additional charged allowed  :  2
additional neutrals allowed :  2

Possible background channels (sigma_tot = 10632.820 mub) are:
-------------------------------------------------------------------------------
  n  |  idx  |  mode                             |  xs [mub]  |  p_thres [GeV/c]
-------------------------------------------------------------------------------
  1  |    6  |  2pi+ 2pi- 2pi0                   |   5368.798 |      0.000
  2  |    5  |  2pi+ 2pi- pi0                    |   2514.088 |      0.000
  3  |   30  |  pi+ pi- 2pi0                     |   1223.618 |      0.000
  4  |   19  |  pb p pi+ pi- pi0                 |    916.475 |      1.604
  5  |   35  |  K+ K- pi+ pi- pi0                |    367.190 |      0.000
  6  |    3  |  pi+ pi- pi0                      |    151.435 |      0.000
  7  |   40  |  pi+ pi- KS KL pi0                |     91.215 |      0.000


EvtGen decay file
------------------
# Possible backgrounds for pi+ pi- pi0  @  E_cm = 3.100 GeV / p_pbar =  4.076 GeV/c
# Total cross section = 10632.820 mub

Decay pbarpSystem0
  0.5049  pi+ pi+ pi- pi- pi0 pi0                    PHSP; # x-sec =  5368.798 mub
  0.2364  pi+ pi+ pi- pi- pi0                        PHSP; # x-sec =  2514.088 mub
  0.1151  pi+ pi- pi0 pi0                            PHSP; # x-sec =  1223.618 mub
  0.0862  anti-p- p+ pi+ pi- pi0                     PHSP; # x-sec =   916.475 mub
  0.0345  K+ K- pi+ pi- pi0                          PHSP; # x-sec =   367.190 mub
  0.0142  pi+ pi- pi0                                PHSP; # x-sec =   151.435 mub
  0.0086  pi+ pi- K_S0 K_L0 pi0                      PHSP; # x-sec =    91.215 mub
Enddecay
End
```

Insertion 2: Number of Events to Simulate

- Signal channel with decays involving BR_i (product = f_{BR})
- Simulate S_0 signal and B_0 background events

- Question: What is good choice for S_0 and B_0?
  - S_0 depends on desired relative efficiency uncertainty $\Delta\varepsilon_{S,rel}$
  - B_0 → input: σ_S, σ_B, ε_S, BR_i, desired $\Delta B/B<\Delta_{rel}$ and $S_d/B_d>r$ in data

$$\frac{S_d}{B_d} = \frac{\mathcal{L}\cdot\sigma_S\cdot f_{BR}\cdot\varepsilon_S}{\mathcal{L}\cdot\sigma_B\cdot\varepsilon_B} > r\,, \quad \varepsilon_B = \frac{B}{B_0}\,, \quad \frac{\Delta B}{B} = \frac{1}{\sqrt{B}} < \Delta_{rel}$$

$$\varepsilon_B < \frac{\sigma_S\cdot f_{BR}\cdot\varepsilon_S}{\sigma_B\cdot r}\,, \quad B_0 > \frac{1}{\varepsilon_B\cdot\Delta_{rel}^2} = \frac{\sigma_B\cdot r}{\sigma_S\cdot f_{BR}\cdot\varepsilon_S\cdot\Delta_{rel}^2}$$

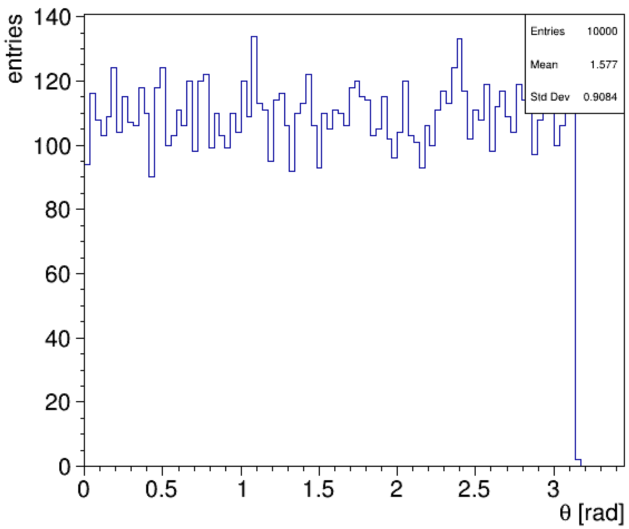
- Example:
  σ_S=100nb, σ_B=10mb, ε_S=20%, f_{BR}=2%, Δ_{rel}=10%, r=3

  → bkg suppression $\varepsilon_B < 1.33\cdot 10^{-8}$, events $B_0 > 7.5\cdot 10^9$ events,

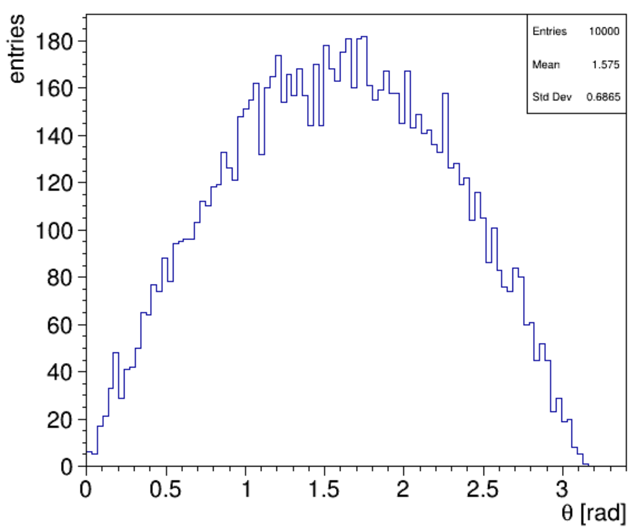
Box Generator - The Particle Gun

- **Box Generator -** Acceptance and resolution studies
  - Single particles of defined species and multiplicity
  - Setting ranges in p, pt, θ, cos(θ), φ, y, η
  - Setting fixed arbitrary origin or from box volume

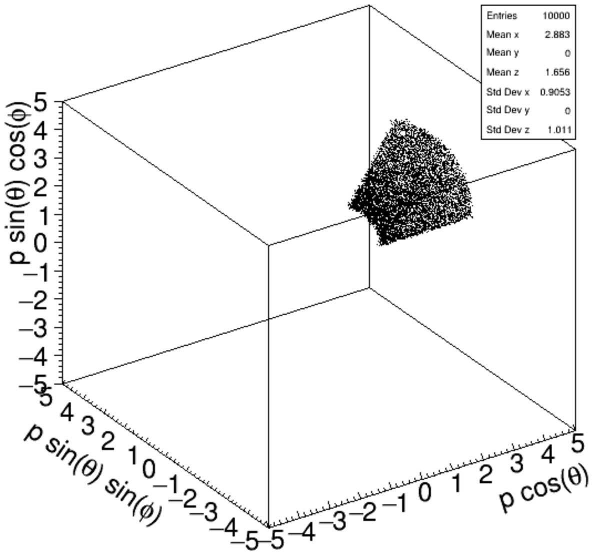
uniform θ



uniform cos(θ)



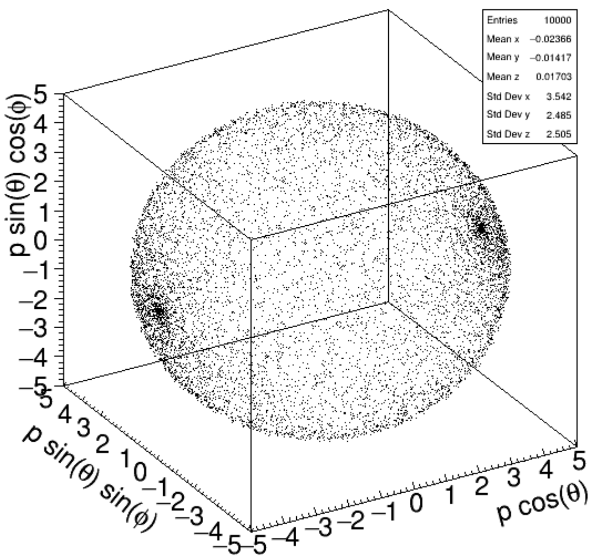
p, θ - range, fixed φ



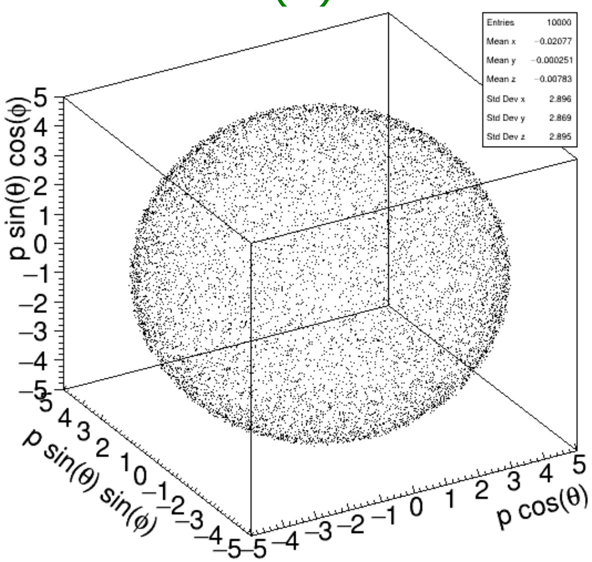
Box Generator - The Particle Gun

- **Box Generator -** Acceptance and resolution studies
  - Single particles of defined species and multiplicity
  - Setting ranges in p, pt, θ, cos(θ), φ, y, η
  - Setting fixed arbitrary origin or from box volume

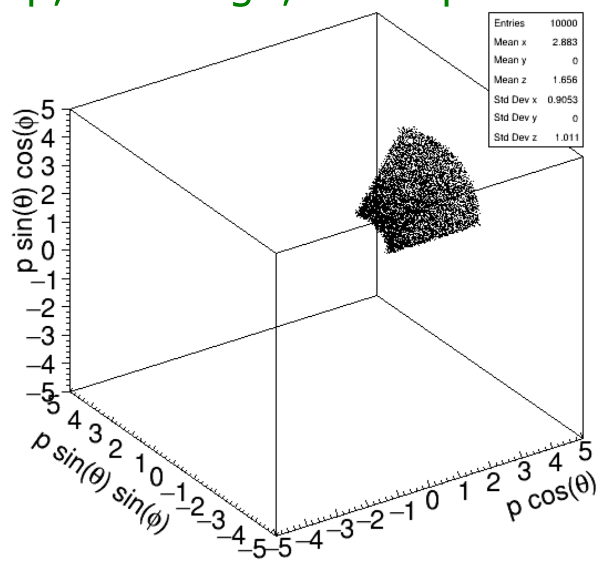
uniform θ



uniform cos(θ)



p, θ - range, fixed φ



GENERATION AND SIMULATION

Generation and Simulation

- Three levels of studying particle distrib. with different purpose

- Stand-alone usage of generators (EvtGen, DPM, FTF)
  - Study undistorted true distributions
  - Principle phase space/multiplicity considerations

- Fast Simulation
  - Simplified effective simulation of acceptance & resolution
  - High speed, easy configurable for physics/detector performance

- Full Simulation
  - Full Geant featured microscopic particle transport
  - Realistic digis and reconstruction simulation

EvtGen - Stand-alone Usage

- Stand-alone executable: `build/bin/simpleEvtGen(RO)`
  - ASCII (`simpleEvtGen`) or ROOT (`simpleEvtGenRO`) output
- Needs links/copies of `evt.pdl` and `DECAY.DEC` to directory
- Call w/o parameter prints help text

```
build/bin > ./simpleEvtGenRO

USAGE:
simpleEvtGen <particle> <dec-file> <# events> <pbar-mom/cms-energy> <rand seed> <A_Target>

  <particle>  = particle type to decay, e.g. 'eta_c', 'pbarpSystem' etc.
  <dec-file>  = EvtGen decay file (.DEC) to use; see directory 'test' for examples
  <# events>  = number of events to produce; default value = 10
  <pbar-mom>  = (>0) momentum of the pbar beam; (<0) negativ cms energy;
                 default value = mass of <particle>, mandatory, when <particle> = pbarpSystem
  <rand seed> = random seed for TRandom3. Value < 0 = use default random gen.; default = -1
  <A_Target>  = target nucleus mass number; mandatory when <particle> = 'pbarASystem'

build/bin > ./simpleEvtGenRO pbarpSystem Jpsi2pi.dec 1000 7.0
...
build/bin > root -l evtOutput.root
root [0] .ls
TFile**         evtOutput.root
 TFile*         evtOutput.root
  KEY: TTree    ntp;1   ntp
```

EvtGen - ROOT Output

- Contents of the output TTree are event based arrays with

Branches	Content
`ev, nTrk`	Event, #particles in event
`N, Id`	particle ID, pdg code
`M1, M2, nDau, DF, DL`	Mother - daughter information
`E, px, py, pz, pt, p, tht, m`	4-vector information
`t, x, y, z`	space-time vertex

arrays

- Array indices according order in decay file configuration
- For first example

DF[0] DL[0] DF[1]

  – 0 = pbarpSystem, 1 = J/psi, 2 = pi+, 3 = pi-, 4 = mu+,...

```
build/bin > root -l evtOutput.root
root [0] ntp->Draw("m[1]") // plots generated mass distribution of J/psi
```

EvtGen - ROOT Output

- Contents of the output TTree are event based arrays with

arrays

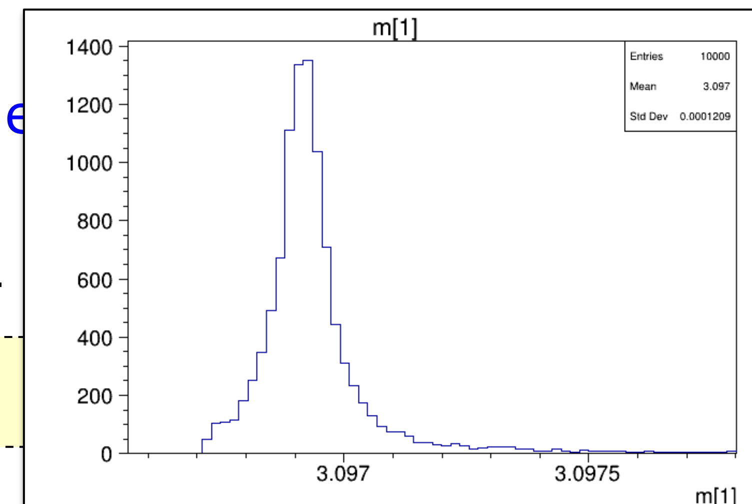
Branches	**Content**
`ev, nTrk`	Event, #particles in event
`N, Id`	particle ID, pdg code
`M1, M2, nDau, DF, DL`	Mother - daughter information
`E, px, py, pz, pt, p, tht, m`	4-vector information
`t, x, y, z`	space-time vertex

- Array indices according order in de
- For first example

DF[0]

  – 0 = pbarpSystem, 1 = J/psi, 2

```
build/bin > root -l evtOutput.root
root [0] ntp->Draw("m[1]") // plots generated
```



DPM - Stand-alone Usage

- Stand-alone executable: `build/bin/DPMGen`
- Input parameters:
  - Random seed
  - Beam momentum (for PANDA: 1.5 ... 15 GeV/c)
  - Mode (0 = inelastic, 1 = inel. + elastic, 2 = elastic)
  - Un/stable settings (e.g. '111 0' lets π^0 be decayed by DPM)
  - Number of events
- Output:
  - ROOT file containing TTree 'data' with TParticles

DPM - ROOT Output

- Relevant (and filled) information in ROOT output

Branches/Methods	Content
Npart	#particles in event
fPdgCode	PDG code of particle
fE, fPx, fPy, fPz, fCalcMass	4-vector information
Theta(), Phi(), Pt(), P() Eta(), Y()	Additional kinematic information

arrays

- No mother-daughter relations stored

```
build/bin > ./DPMGen
Give as seed a large float number (eg. 123456.): 23
 Enter  P_lab(GeV/c), 15
 Enter  Elastic : 0., 1. or 2.
0
Enter:  particle PDGcode and status
   (for example -- Pi0 unstable:  111  0), To go to event generation, Enter:  0  0
0 0
 Enter  N_Events 30000

build/bin > root -l Background-micro.root
root [0] data->Draw("fCalcMass","fabs(fPdgCode)==3122")
```

DPM - ROOT Output

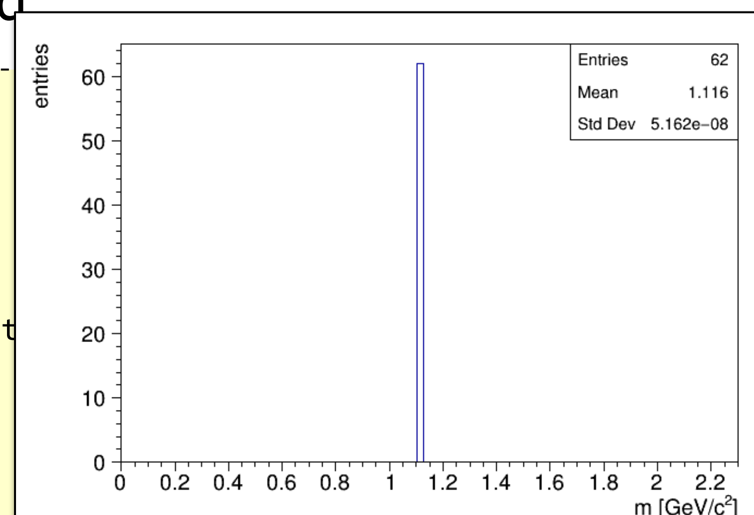
- Relevant (and filled) information in ROOT output

arrays

Branches/Methods	Content
Npart	#particles in event
fPdgCode	PDG code of particle
fE, fPx, fPy, fPz, fCalcMass	4-vector information
Theta(), Phi(), Pt(), P() Eta(), Y()	Additional kinematic information

- No mother-daughter relations stored

```
build/bin > ./DPMGen
Give as seed a large float number (eg. 123456.): 23
 Enter  P_lab(GeV/c), 15
 Enter  Elastic : 0., 1. or 2.
0
Enter:  particle PDGcode and status
   (for example -- Pi0 unstable:  111  0), To go to event
0 0
 Enter  N_Events 30000

build/bin > root -l Background-micro.root
root [0] data->Draw("fCalcMass","fabs(fPdgCode)==3122")
```



Fast Simulations - Motivation

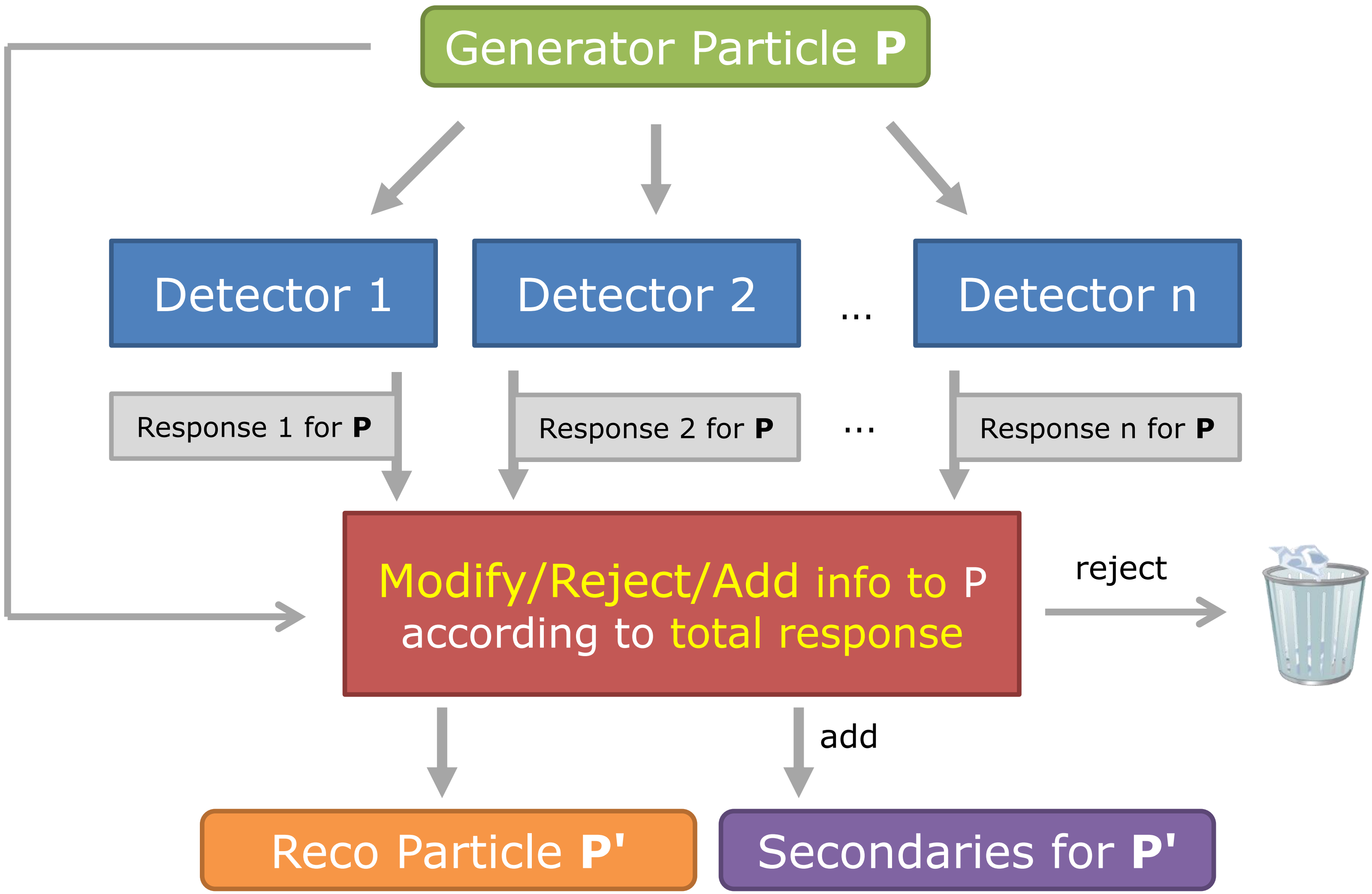
- Instead of microscopic simulation with Geant propagation
  - Effective smearing and rejetion of generator particles

- Purpose: Study of
  - Reaction kinematics
  - Detector configurations

- Simulate large number of events in short time
- Well controlable to anticipate future detector performance
- Same analysis code as for full simulation and reconstruction

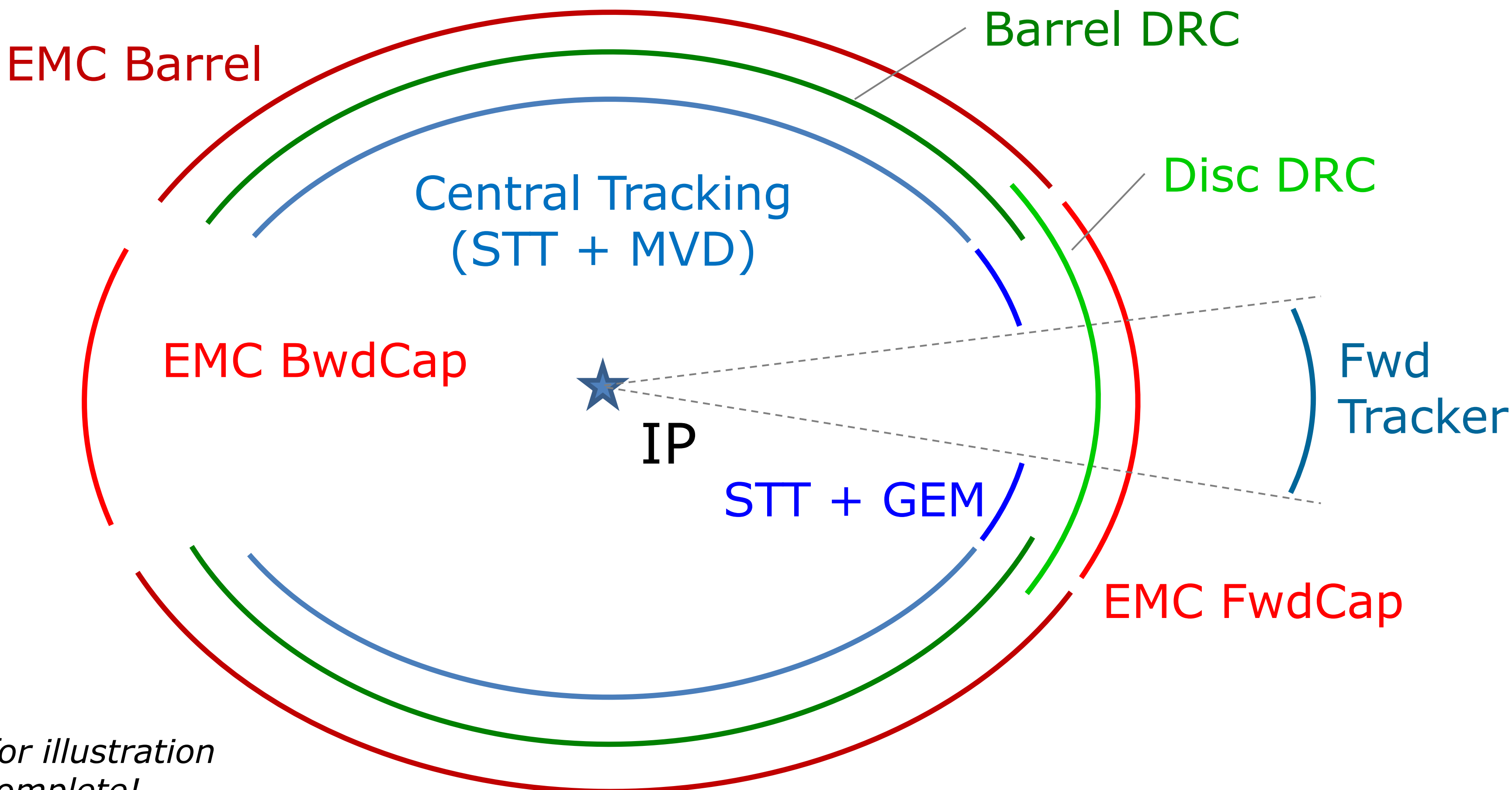
- Goal is not necessarily to imitate current full simulation performance, in particular where this is not yet final

FastSim - Concept



FastSim - Acceptance Modelling

- Spatial acceptance typically defined by polar θ range from IP
- Different trackers or EMCs should not overlap!



Just for illustration
Not complete!

FastSim - Detector Modules

	Target Spectrometer	Forward Spectrometer
TRK	STT; STT + MVD + GEM; STT + MVD; MVD + GEM; MVD Vtx	Fwd TS
EMC	EMC Barrel; EMC Fwd; EMC Bwd	EMC FS
PID	EMC Barrel; STT PID; DIRC PID; EMC Fwd; MVD PID; DISC PID; EMC Bwd; MUO PID	EMC FS PID; RICH PID; MUO FS PID

FastSim - Detector Parameters/Response

- Detector parameters
  - Global detection efficiency ε
  - Spatial acceptance in θ (sometimes φ)
  - Kinematic acceptance (p_{min}, $p_{t,min}$, E_{min}, ...)
  - Resolutions: dp/p, dE/E, dt/t, $d\theta$, $d\varphi$, vertex
  - Detector specific parameters (geometry, # layers, granularity,...)
- Detector responses
  - Accept/reject particle *(no or pure PID response $\rightarrow$ rejection)*
  - Current resolution dp/p, dE/E, dt/t, $d\theta$, $d\varphi$, vertex
  - PID probabilities (P_e, P_μ, P_π, P_K, P_p)
  - Raw PID information (E_{EMC}, θ_C, dE/dx, L_{iron}, ...)
- Total response
  - created from individual responses + applied to particle

FastSim - Free Detector Configuration

→ Simple change of detector parameters on macro level

```
fSim->AddDetector("ScSttAlone",
      "thtMin=145.0 thtMax=159.5 ptmin=0.1 pRes=0.04 thtRes=0.006 phiRes=0.007 efficiency=0.25" );

fSim->AddDetector("ScSttMvd",
      "thtMin=20.9  thtMax=145.0 ptmin=0.1 pRes=0.02 thtRes=0.001 phiRes=0.001 efficiency=0.85" );

fSim->AddDetector("ScSttMvdGem",
      "thtMin=7.8   thtMax=20.9  ptmin=0.1 pRes=0.02 thtRes=0.001 phiRes=0.001 efficiency=0.85" );

fSim->AddDetector("ScMvdGem",
      "thtMin=5.0   thtMax=7.8   ptmin=0.1 pRes=0.03 thtRes=0.001 phiRes=0.001 efficiency=0.60" );
...

fSim->AddDetector("EmcFwCap",  "thtMin=10.0  thtMax=22.0  Emin=0.01 dist=2.5" );
fSim->AddDetector("EmcBarrel", "thtMin=22.0  thtMax=142.0 Emin=0.01 barrelRadius=0.5" );
fSim->AddDetector("EmcBwCap",  "thtMin=142.0 thtMax=160.0 Emin=0.01 dist=0.7" );
...

fSim->AddDetector("DrcBarrel", "thtMin=22.0 thtMax=140.0 dthtc=0.01 nPhotMin=5 effNPhotons=0.075" );
fSim->AddDetector("DrcDisc",   "thtMin=5.0  thtMax=22.0  dthtc=0.01 nPhotMin=5 effNPhotons=0.075" );
...
```

Running Fast Simulation

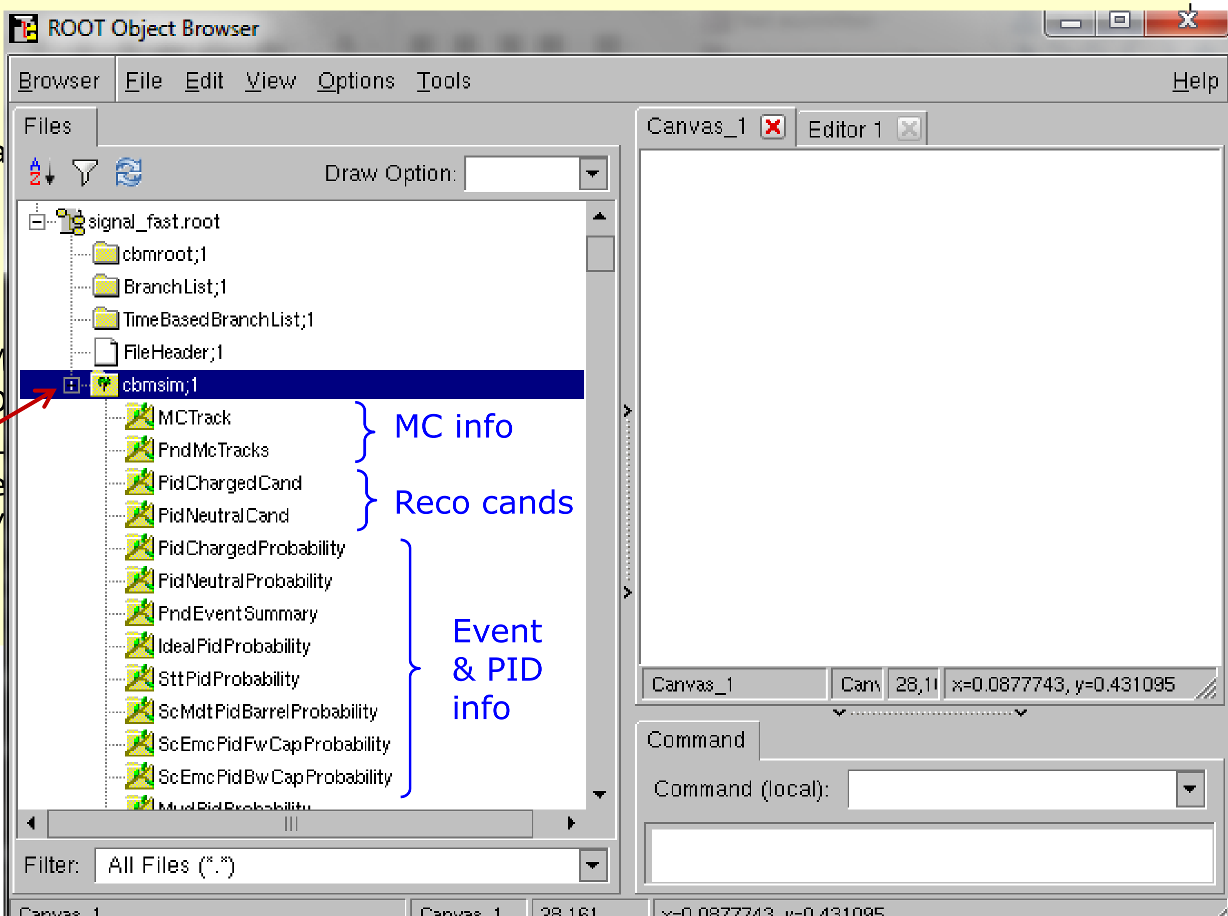
- Macro tutorials/thailand2017/tut_fastsim.C
- Creates same output format as Full Simulation

```
> root -l -b -q 'tut_fastsim.C(1000,"signal","pp_jpsi.dec", 7.0, "pbarpSystem")'

> root -l signal_fast.root

root [0]
Attaching file signal_fast.root as _file0...
(TFile *) 0x385f240
root [1] .ls
TFile**         signal_fast.root
 TFile*         signal_fast.root
  KEY: TFolder  cbmroot;1       Main Folder
  KEY: TList    BranchList;1    Doubly linked list
  KEY: TList    TimeBasedBranchList;1   Doubly linked list
  KEY: FairFileHeader   FileHeader;1
  KEY: TTree    cbmsim;1        /cbmroot

root [1] TBrowser b
```

Running Fast Simulation

- Macro tutorials/thailand2017/tut_fastsim.C
- Creates same output format as Full Simulation

```
> root -l -b -q 'tut_fastsim.C(1000,"signal","pp_jpsi.dec", 7.0, "pbarpSystem")'

> root -l signal_fast.root

root [0]
Attaching file signal_fast.root a
(TFile *) 0x385f240
root [1] .ls
TFile**         signal_fast.root
 TFile*         signal_fast.root
  KEY: TFolder  cbmroot;1         M
  KEY: TList    BranchList;1      D
  KEY: TList    TimeBasedBranchLi
  KEY: FairFileHeader   FileHeade
  KEY: TTree    cbmsim;1          /

root [1] TBrowser b
```



EXERCISES

Exercises Preparations

- Preparation/hints in `tutorials/thailand2017/README`

- To use the event generators stand-alone, do:

```
cd $VMCWORKDIR/tutorials/thailand2017
. ./prep_generators
```

- This creates some links and copies files to ./generators

```
mkdir generators
ln -s $FAIRLIBDIR/../bin/simpleEvtGenRO generators/
ln -s $FAIRLIBDIR/../bin/DPMGen generators/
ln -s $FAIRLIBDIR/../bin/FTFGen generators/
cp $VMCWORKDIR/pgenerators/EvtGen/EvtGen/Private/evt.pdl generators/
cp $VMCWORKDIR/pgenerators/EvtGen/EvtGen/Private/DECAY.DEC generators/
cp $VMCWORKDIR/pgenerators/FtfEvtGen/Pbar*.mac generators/
```

- Change to ./generators to run them

Exercises Suggestions

- Rho Tutorial website:
  http://panda-wiki.gsi.de/cgi-bin/view/Computing/PandaRootRhoTutorial
- Take a look to `tutorials/thailand2017/README`
  `Exercises: #7 - #9`

1. Write decay file for an arbitrary channel (or use existing)
2. Generate some stand-alone EvtGen events
3. Generate some background events with DPM at same energy
4. Compare momentum distributions of final states
5. Run FastSimulation with same inputs
6. Compare stand-alone and fast simulation output

Exercises Preparation

Preparation/hints in `tutorials/thailand2017/README`

- **FIRST setup environment: `source ~/workshop/build/PandaRoot_trunk.../config.sh`**

- To have some data for tutorial macros, do one of the following

```
a) . ./tut_runall.sh 1000   # sim/reco 1000 events pbarp -> J/psi pi+ pi-
b) cp data/signal_p*root .  # preproduced default data
```

- Macros named 'tut_ana...C' are stubs and should be completed by you.

- At places marked with ' #### EXERCISE: ...' some code needs to be added;

  Run macros (default or different input data) with

```
> root -l tut_ana...C                 # signal_pid.root, signal_par.root
> root -l 'tut_ana...C(0,"mydata")'   # mydata_pid.root, mydata_par.root
```

- If getting stuck, sample solutions are in the subfolder 'solution'

  Run solution macros directly (default or different input data) with

```
> root -l solution/tut_ana...C
> root -l 'solution/tut_ana...C(0,"mydata")'
```